Remarks

Entry of the amendments, reconsideration of the application, as amended, and allowance of all pending claims are respectfully requested. Upon entry of the amendments, claims 45-64 are pending.

With the above amendments, applicant is further clarifying aspects of his invention. For instance, applicant is explicitly reciting that the automatically restoring is in response to modifying source code of the program, and that the automatically restoring comprises comparing one or more attributes of an instruction profile created based on a first version of the source code with one or more attributes of machine instructions generated for a second version of the source code to determine the appropriate location to automatically restore the breakpoint. Support for these amendments can be found throughout applicant's specification. For instance, support may be found in paragraph 27 on page 8, and pages 10-17 of applicants' specification, as well as in FIGs. 3-5b. Thus, no new matter is added.

In a final Office Action, dated May 19, 2005, previously pending claims 1-11, 13-22, 24, 27-39 and 41-44 were rejected under 35 U.S.C. 102(e) as being anticipated by Olsen (U.S. Patent No. 6,263,489). Further, previously pending claims 12 and 40 were rejected under 35 U.S.C. 103(a) as being unpatentable over Olsen in view of Davidson (U.S. Patent No. 5,819,093). Applicant respectfully, but most strenuously, traverses these rejections to any extent deemed applicable to the amended claims.

In one aspect, applicant's invention is directed to automatically restoring debugging breakpoints subsequent to modifying the source code of a program. The automatically restoring restores a breakpoint to the same step the breakpoint was set prior to modification of the source code, although that step is now at a different location within the modified program.

As an example, to automatically restore the debugging breakpoint to the same step of the program that had the breakpoint prior to modification, the location (e.g., line of code) that includes that selective step is determined. This determination is made by creating an instruction profile that includes attributes of a number of generated instructions that are associated with the selected step. These attributes include attributes of the instruction (such as, an operation code, operands and operation type obtained from reading the instruction) of the selected step, and possibly attributes of other instructions in proximity to the selected step. This is described further with reference to the following program:

Source View On Debugger Front End	**Machine Instructions In Application**
Line 1: /*Test program 1*/	00001 L 1,4,0
Line 2: int i = 0;	00002 L 1,5,1
Line 3: int j = 1;	00003 TC 1,4,5
Line 4: if (i > j) {	00004 BL 6
Line 5: print hello;	00005 P hello
Line 6: }	00006 TC 1,4,5
Line 7: if (j > i) {	00007 BH 9
Line 8: print world;	00008 P world
Line 9: }	00009 G back
Line 10: exit;	

In this example, a line breakpoint is set for line 7 in the Source View, which corresponds to line 6 of the Machine Instructions. However, to describe line 6, which looks like line 3, attributes of line 7 of the Machine Instructions are also included in the instruction profile in order to uniquely identify line 6 as the line in which the breakpoint is associated with. The number of instructions to be included in the instruction profile is determined based on a calibration technique described in applicant's specification.

When the program is modified, the attributes of the instruction profile are compared to the attributes of the newly generated instructions to determine where the particular line of code of interest (e.g., line 6) has moved in the program. The various comparisons made to different instructions within the modified program yield difference counters. In this example, the difference counter having the smallest value indicates the location of the selected step. Thus, the debugger automatically restores the breakpoint to that step.

In one particular example, applicant claims a method of restoring debugging breakpoints (e.g., independent claim 1). The method includes, for instance, having a breakpoint that is set to a selected step of a first version of source code of a program, the program being absent embedded debug commands; creating an instruction profile for the selected step, the instruction profile including one or more attributes of one or more machine instructions generated for the selected step and one or more attributes of zero or more other machine instructions generated for the first version of source code; and automatically restoring the breakpoint to the selected step of a modified program, in response to modification of the first version of source code to provide the modified program having a second version of source code, wherein the selected step is at a different location within the modified program, wherein the automatically restoring includes comparing one or more attributes of one or more machine instructions generated for the second version of source code with one or more attributes of the instruction profile created based on the first version of source code to determine the different location.

Thus, in this aspect of applicant's claimed invention, a breakpoint is automatically restored to the selected step of a modified program, in response to modification of the source code of the program. Further, the restoring compares attributes of one or more machine instructions generated for a second version of source code with one or more attributes of an instruction profile created based on a first version of source code to determine the new location for the breakpoint. These features are very different from the teachings of Olsen.

In Olsen, while a technique is described for debugging optimized code, Olsen fails to describe, teach or suggest one or more aspects of applicant's claimed invention. For instance, there is no description, teaching or suggestion in Olsen of automatically restoring a breakpoint, in response to modification of a first version of the source code to provide a modified program having a second version of the source code. There is no description in Olsen of modifying the source code. Instead, Olsen is directed to problems associated with optimized code and does not address handling different versions of source code. Specifically, there is no description, teaching or suggestion in Olsen of automatically restoring the breakpoint to the selected step of a modified program, in response to modification of the first version of source code. Again, Olsen is silent as to modifying source code. Instead, Olsen is concerned with optimized machine code. Since Olsen fails to describe, teach or suggest at least applicant's claimed element of automatically restoring the breakpoint to the selected step of a modified program, in response to modification of the first version of source code, applicant respectfully submits that Olsen does not anticipate or render obvious applicant's claimed invention.

As a further example, Olsen fails to describe, teach or suggest applicant's claimed element of comparing one or more attributes of one or more machine instructions generated for the second version of the source code with one or more attributes of the instruction profile created based on a first version of the source code to determine the location in which to restore the breakpoint after modification of the source code. Again, Olsen is directed to the debugging of optimized code and does not address automatically restoring breakpoints after source code has been modified from one version to another version. Further, there is no description, teaching or suggestion in Olsen of comparing attributes from one version of source code to attributes from a second modified version of source code. There is no description, teaching or suggestion in Olsen of making such a comparison to determine the location in which to automatically restore the breakpoint, subsequent to modification of the source code. In Olsen, the tables that are created are for that one version of source code. There is no description, teaching or suggestion of making comparisons based on the two different versions of source code, as claimed by applicant. Since Olsen is directed to debugging optimized code and is not directed to restoring breakpoints after the source code has been modified, and since Olsen fails to describe, teach or suggest applicant's claimed comparing, applicant respectfully submits that independent claim 1 is patentable over Olsen.

For at least the above reasons, applicant respectfully submits that his invention, as recited in the now pending independent claims, is patentable over Olsen. Further, the cited art of record does not overcome the deficiencies of Olsen.

Additionally, applicant respectfully submits that the dependent claims are patentable for the same reasons as the independent claims, as well as for their own additional features. For instance, dependent claims 46, 54 and 60 explicitly recite that the comparing compares one or more operation codes of the one or more machine instructions generated for the second version of source code with one or more operation codes of the instruction profile (generated based on the first version of source code) to determine which machine instruction of the modified program corresponds most closely to the selected step. This is not described, taught or suggested in Olsen. There is no mention in Olsen of different versions of source code and there is no description, teaching or suggestion in Olsen of comparing operation codes of machine instructions generated from different versions of source code to determine where the selected breakpoint should be placed. Since this is missing from Olsen, applicant respectfully submits that Olsen does not anticipate or render obvious applicant's claimed invention, as claimed in dependent claims 46, 54 and 60.

As a further example, applicants explicitly recite in dependent claims 47, 55 and 61that the instruction profile includes a source line number for the selected step and a length of the first version of source code and that the automatically restoring uses these values to determine a starting point within the modified program to select the one or more instructions generated for the second version to be used in the comparing. Again, this is not described, taught or suggested in Olsen. In Olsen, high water marks and low water marks are used to determine which lines of code are to be emulated. However, there is no description, teaching or suggestion of using the source line number of a first version of the source code and the length of the first version of source code to determine a starting point from which one or more instructions generated for the second version of source code are to be used in the comparing in order to automatically restore the breakpoint to the selected step within the modified program. Thus, applicant respectfully submits that dependent claims 47, 55 and 61 are patentable over Olsen.

In yet a further example, applicant recites in dependent claims 51, 58 and 64 a stepwise procedure for choosing a number of instructions to be included in the instruction profile used in the comparing step. Applicants respectfully submit that this stepwise procedure, which includes, for instance, selecting a number of instructions to be included in a calibration profile; generating the calibration profile for a chosen line of the program, the calibration profile having the selected number of instructions for the chosen line; comparing one or more attributes of the calibration profile to one or more attributes of at least one line of code of the program to obtain a result; determining whether the result is an unambiguous result; and repeating, zero or more times, the selecting, the generating, the comparing, and the determining until the determining indicates an unambiguous result, wherein the selected number of instructions increases at each iteration, and wherein the selected number of instructions indicates, when there is an indication of an unambiguous result, the number of machine instructions to be included in the instruction profile, is not described, taught or suggested in Olsen.

Based on the foregoing, applicant respectfully submits that his invention, as claimed, is patentable over Olsen and any other cited art. Thus, applicant respectfully requests an indication of allowance of all pending claims.

Should the Examiner wish to discuss this case with applicant's attorney, please contact applicant's attorney at the below listed number.

Respectfully submitted,

Blanche E. Schiller

Blanche E. Schiller
Attorney for Applicants
Registration No.: 35,670

Dated: September 19, 2005.

HESLIN ROTHENBERG FARLEY & MESITI P.C.
5 Columbia Circle
Albany, New York 12203-5160
Telephone: (518) 452-5600
Facsimile: (518) 452-5579